ISSN 1561-5359. Штучний інтелект, 2016, № 1

УДК 004.27

І.А. Клименко
Національний технічний університет України «Київський політехнічний інститут»
проспект Перемоги, 37, м. Київ, Україна, 03056

ФОРМАЛІЗАЦІЯ АДАПТИВНОГО ВІДОБРАЖЕННЯ ЗАДАЧ У РЕКОНФІГУРОВАНИХ ОБЧИСЛЮВАЛЬНИХ СИСТЕМАХ НА ПЛІС

I.A. Klymenko
National Technical University of Ukraine "Kyiv Polytechnic Institute"
37, Prospect Peremohy, Kyiv, Ukraine, 03056

THE FORMALIZATION OF ADAPTIVE TASKS MAPPING IN THE RECONFIGURABLE COMPUTING SYSTEMS ON FPGAS

Розроблено математичні моделі адаптивної реконфігурації, що визначають значення критеріїв ефективності реконфігурованих обчислень. Запропоновано новий підхід до скорочення критичного часу виконання паралельних алгоритмів за рахунок видалення непродуктивної складової часу реконфігурації з критичного шляху графу алгоритму. Розроблено та досліджено програмну модель запропонованих засобів адаптивного відображення алгоритму на реконфігуровану обчислювальну структуру на ПЛІС.
Ключові слова:реконфігуровані обчислювальні системи, накладні видатки реконфігурації, динамічна реконфігурація, ПЛІС.

Formal models of adaptive reconfiguration were developed. It allowed determining the value of the performance criteria of reconfigurable computing system. A new approach of reducing the critical execution time of parallel algorithms was proposed by removing reconfiguration time overheads from the critical path of algorithm`s graph. Program model of proposed means for adaptive tasks mapping on reconfigurable FPGA computing structure.
Keywords: reconfigurable computer systems, reconfiguration overhead, dynamic reconfiguration, FPGA.

Застосування програмованих логічних інтегральних схем (ПЛІС) як елементної бази для побудови обчислювального середовища дозволяє реалізувати обчислювальну систему практично будь-якої складності з гнучкою архітектурою. Можливість перепрограмування кристалу на фізичному рівні забезпечує доволі швидке налаштування обчислювальної структури на реалізацію довільних алгоритмів. Ця технологія лежить в основі парадигми реконфігурованих обчислень [1 – 3], що на сьогодні є актуальним напрямком підвищення ефективності паралельних обчислювальних систем. Така парадигма дозволяє, з одного боку, забезпечити лінійний приріст користувацької ефективності, а з іншого – забезпечити економічну привабливість і широку функціональність.

В останні роки поява ПЛІС, що динамічно реконфігуруються, створила передумови і нові можливості для підвищення ефективності паралельних обчислювальних систем за рахунок значного зменшення накладних витрат часу на перепрограмування кристалу ПЛІС, а також можливості реконфігурації обчислювальної структури в динамічному режимі (*RunTime*). Але відомі технології та методи відображення алгоритмів, які розроблені для фіксованих або статично реконфігурованих обчислювальних середовищ, не можуть ефективно використовуватись в динамічно реконфігурованих обчислювальних системах, з огляду на їх функціональні особливості та апаратні обмеження. Реконфігуровані обчислювальні системи мають ряд обмежень, які стосуються часу обчислення, що

обумовлено непродуктивними витратами часу, продуктивності й енергоресурсів під час реконфігурації обчислювальної структури. Важливою є проблема обмеження апаратних ресурсів ПЛІС.

У зв'язку з цим, дослідження, виконані в статті, представляють науковий і практичний інтерес. У роботі вирішується актуальна проблема розробки засобів адаптивного відображення алгоритмів на обчислювальне середовище в системах, що характеризуються певними функціональними та апаратними обмеженнями і змінними умовами відображення, зумовленими використанням елементної бази, що динамічно перепрограмується.

Огляд відомих рішень

Відомо багато методів і технологій відображення задач в реконфігурованих обчислювальних системах, які одночасно реалізують механізми зменшення накладних витрат реконфігурації. Найбільш характерні серед них: повторне використання ресурсів [1], кешування конфігураційних даних [2], випереджувальна реконфігурація [3]. Але більшість відомих механізмів відображення реалізовані на рівні програмного забезпечення надбудови операційної системи, подолання просторових обмежень ПЛІС здійснюється стандартними засобами, наприклад, дефрагментацією обчислювальної поверхні ПЛІС [1, 2], вивантаженням некритичних конфігурацій [1, 3], відмовою виконання задач [1, 2, 3]. Все це вносить додаткові накладні витрати в процес реконфігурованих обчислень та підвищує часову складність алгоритмів відображення. Такі методи найбільш ефективні для реалізації статичних алгоритмів відображення. У динамічно реконфігурованих обчислювальних системах виникає необхідність певної модифікації відомих засобів для вирішення проблеми ефективного відображення алгоритмів на динамічно реконфігуроване обчислювальне середовище.

Постановка задачі

Визначимо основні критерії ефективності реконфігурованих обчислювальних систем, на підставі показника прискорення [3].

$$\rho = \frac{T_{SW}}{T_{REC} + T_{HW}}.$$

Показник прискорення дорівнює відношенню часу обчислення задачі програмними засобами T_{SW} до суми часу апаратного виконання завдання і часу реконфігурації обчислювальної області кристала ПЛІС $(T_{REC} + T_{HW})$, відповідно. Під часом реконфігурації ми маємо на увазі час передачі конфігураційних даних і безпосередньо час, витрачений на перепрограмування певної обчислювальної області на кристалі ПЛІС.

Однак, зазначений показник прискорення орієнтований на використання в статичних реконфігурованих обчислювальних системах [4]. Для динамічно реконфігурованих систем його застосування неефективне, оскільки він не враховує додаткові витрати часу, обумовлені високою складністю реалізації процесів планування реконфігурованих ресурсів та відображення завдань, що пов'язано з необхідністю урахування фізичних параметрів і обмежень кристалів ПЛІС та пов’язаними з цим змінюваними умовами відображення. З точки зору реалізації динамічного відображення задач, у статті запропоновано модифікувати відомий

показник прискорення, з урахуванням часової складності процесів планування і організації реконфігурованих обчислювальних ресурсів ($T_{CONTROL}$):

$$\rho = \frac{T_{SW}}{[T_{CONTROL} + T_{REC}] + T_{HW}},$$

Згідно з модифікованим показником прискорення, основним критерієм ефективності динамічно реконфігурованих обчислювальних систем є час реконфігурованих обчислень, який складається з часу реконфігурації обчислювальної структури $(T_{CONTROL} + T_{REC})$ та часу обчислення на ПЛІС T_{HW}. Зменшення часу обчислень у реконфігурованих обчислювальних системах може бути досягнуто за рахунок забезпечення оптимальної відповідності обчислювального алгоритму і структури реконфігурованого обчислювального середовища, що детально розглянуто авторами в попередній роботі [5].

Залежність функції мінімізації часу відображення від затримок реконфігурації виражатися наступний чином:

$$\min(T_{MAPP}) = \min(T_{CONTROL}) + \min(T_{COMM}) + T_{CONFIG}.$$

Перемістивши процес відображення завдань з рівня операційної системи на локальний рівень абстракції, що розглянуто в роботі [6], виникає можливість об'єктивної оцінки часової складності цього процесу і його оптимізації. Впливати на час конфігурації кристала ПЛІС складно, бо цей час залежить від технічних показників мікросхеми. Однак, функціональні особливості реалізації реконфігурованих обчислювальних систем надають широкі можливості для зменшення комунікаційних затримок під час реконфігурації. На природу виникнення комунікаційних затримок впливають наступні фактори: структура обчислювальної системи; організація адресного простору; структура комунікаційного середовища; місце розташування конфігураційних даних; обсяг конфігураційних даних. Комунікаційні затримки визначають час, витра-чений на процес передавання конфігураційних даних з віддалених бібліотек до інтер-фейсів кристалів ПЛІС, що передує процесу програмування обчислювальної області на поверхні кристала ПЛІС. Цей час визначає непродуктивну складову часу реконфігурації і є критичним критерієм ефективності динамічно реконфігурованих обчислень.

Таким чином, у статті вирішується задача зменшення непродуктивного часу відображення задач на динамічно реконфігуровану обчислювальну структуру на ПЛІС, за рахунок зменшення комунікаційних затримок під час налаштування обчислювального середовища згідно з вимогами кожної вирішуваної в певний момент часу задачі.

Новий підхід до модифікації графа алгоритму ЯПФ

Критичними параметрами обчислювальних алгоритмів, поданих графами ярусно паралельної форми (ЯПФ), які впливають на ефективність обчислень, є критичний шлях і ширина графа алгоритму. Критичний шлях визначає максимальний (критичний) час виконання алгоритму, який згідно з виразом (1) обмежується положенням ($\rho \uparrow \Rightarrow (T_{REC} + T_{HW}) < T_{SW}$). Ширина графа обмежується наявною кількістю обчислювальних ресурсів. Слід зазначити, що всі відомі методи і засоби відображення розроблені для фіксованих обчислювальних структур з необмеженою кількістю обчислювальних ресурсів. При цьому обмеження, якими

характеризуються реконфігуровані обчислювальні системи, роблять неефективними традиційні підходи до модифікації графів алгоритмів з метою скорочення критичного шляху [7]. Наприклад, зменшення критичного шляху за рахунок розширення графа неможливо через апаратні обмеження, а зменшення ширини графа за рахунок збільшення критичного шляху унеможливлюється внаслідок часових обмежень.

Для вирішення розглянутої проблеми в роботі запропоновано методи і засоби зменшення критичного часу виконання алгоритмів у реконфігурованих обчислювальних системах, які засновані на зменшенні непродуктивного часу реконфігурації.

Нехай обчислюваний алгоритм заданий графом ЯПФ $G_M = (V, E)$, де V – множина вершин, а E – множина ребер. У вершинах алгоритму розміщуються макрозадачі, а ребра вказують на залежності між макрозадачами. У контексті вирішення проблеми відображення, для кожної готової до виконання макрозадачі на кристалі ПЛІС шляхом перепрограмування (реконфігурації кристалу) певної області поверхні створюється відповідна обчислювальна структура.

Таким чином, час виконання кожної задачі в вершині графа алгоритму складається з двох складових: часу реконфігурації на реконфігуровану обчислювальну структуру і часу виконання задачі на поверхні кристалу ПЛІС.

У роботі запропоновано новий підхід до модифікації графа алгоритму ЯПФ, метою якого є зменшення критичного шляху. Новий підхід полягає у видаленні з критичного шляху графа ЯПФ алгоритму складової часу відображення завдань, для реалізації чого запропоновано видаляти повторне завантаження конфігураційних даних і здійснювати випереджувальну реконфігурацію функціональних блоків на поверхні ПЛІС.

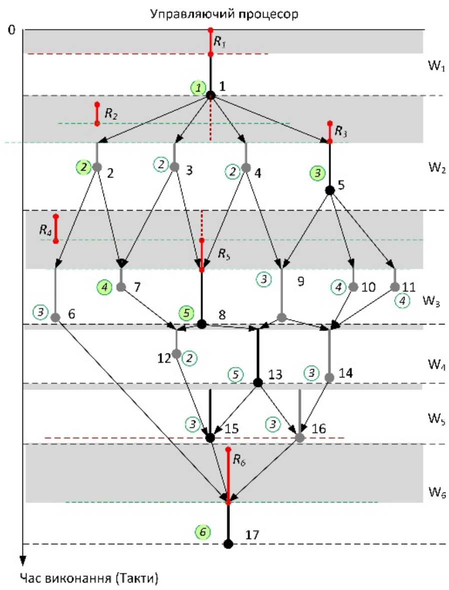


Рис. 1. Повторне використання обчислювальних ресурсів

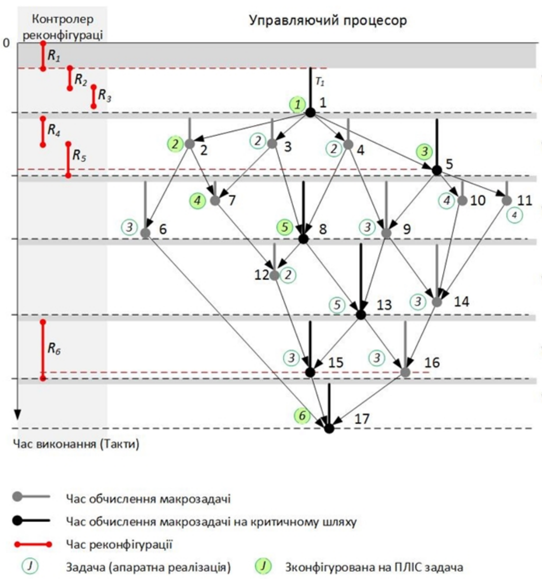


Рис. 2. Випереджувальна реконфігурація

Таким чином, частина складової часу реконфігурації буде видалена з критичного шляху, а частина – переміщена на попередній рівень графа ЯПФ. У результаті такої модифікації графа, критичний шлях буде визначається тільки часом

обчислення задачі. На рис. 1 показано ефект від повторного використання апаратних ресурсів функціональних блоків. На рис. 2 показано, що випереджувальна реконфігурація дозволяє повністю видалити непродуктивний час з критичного шляху графа ЯПФ.

Оцінка часу виконання кожного ярусу графа ЯПФ алгоритму, приклад якого наведено на рис. 2 і рис. 3 під час комплексного застосування розглянутих засобів, наведена в табл. 1.

Таблиця 1. Порівняльна оцінка часу реконфігурації

Номер ярусу	**Стандартна послідовність реконфігурації**	**Видалення повторного завантаження конфігураційних даних**	**Завчасна реконфігурація**
W_1	$T_{W_1} = R_1 + T_1$	$T_{W_1} = R_1 + T_1$	$T_{W_1} = R_1 + T_1$
W_2	$T_{W_2} = 3R_2 + R_3 + T_5$	$T_{W_2} = R_2 + R_3 + T_5$	$T_{W_2} = R_4 + R_5$
W_3	$T_{W_3} = 2R_3 + 3R_4 + T_8$	$T_{W_3} = R_3 + R_4 + T_8$	$T_{W_3} = T_8$
W_4	$T_{W_4} = R_2 + R_5 + R_3 + T_{13}$	$T_{W_4} = T_{12}$	$T_{W_4} = T_{12}$
W_5	$T_{W_5} = 2R_3 + T_{15}$	$T_{W_5} = T_{14}$	$T_{W_5} = R_6$
W_6	$T_{W_6} = R_6 + T_{17}$	$T_{W_6} = R_6 + T_{17}$	$T_{W_6} = T_{17}$

Оцінка часу реконфігурації

Для оцінки часу реконфігурації були визначені і досліджені основні етапи відображення завдань, на підставі чого визначено, що відповідно до місця розміщення вихідних конфігураційних даних можливі три різні послідовності реконфігурації: при розміщенні в віддаленій централізованій бібліотеці конфігурацій у початковому стані (I), при кешуванні конфігурацій у локальній пам'яті обчислювального модуля (II) і безпосередньо на поверхні ПЛІС (III) (рис. 3).



Рис. 3. Основні послідовності завантаження конфігураційних даних на ПЛІС

Виходячи з того, що час виконання алгоритму, поданого графом ЯПФ, визначається його критичним шляхом, отримано математичні вирази для визначення часу його виконання на підставі оцінки критичного часу. У таблиці 2 приведена формальна оцінка часу виконання послідовності задач, що знаходяться на критичному шляху для різних послідовностей реконфігурації з видаленням повторних завантажень конфігураційних даних. Для формальної оцінки часу використано наступні позначення: час виконання впорядкованої послідовності обчислювальних задач (макрооперацій) $[I_j \mid j = \overline{1, K}]$ на критичному шляху B, K –

кількість типів задач, P_j – кількість екземплярів задач, R_j – час завантаження та конфігурування кожної j-ї апаратної задачі на поверхні ПЛІС, T_j – час обчислення кожної задачі. Складова P_jT_j відповідає продуктивному сумарному часу виконання всіх екземплярів апаратної задачі I_j на поверхні реконфігурованої області.

Спосіб адаптивного зменшення критичного часу виконання алгоритму

Скорочення критичного шляху шляхом видалення непродуктивних витрат часу може призвести до того, що інший шлях графа алгоритму стане довшим, ніж критичний. Виникає необхідність послідовного перебору всіх шляхів у пошуках найбільш ефективного рішення. Це – ітераційна задача, яка може бути вирішена швидко при певних умовах, а може затребувати значного часу. Найбільш ефективно це вирішується в статичному режимі. Однак, для реалізації динамічних обчислень, при наявності певних функціональних обмежень, коли умови відображення задач непередбачувані, таке рішення неефективне. У зв'язку з цим, для підвищення ефективності запропонованих рішень, запропоновано новий спосіб поярусного скорочення критичного шляху графа ЯПФ. Запропонований спосіб, на відміну від статичної реалізації [6], заснований на модифікації відомого алгоритму гілок і границь, що запускається в межах кожного ярусу графа ЯПФ алгоритму. Алгоритм заснований на аналізі нащадків кожної вершини в межах певного ярусу графа ЯПФ та запуску для кожного з них відповідної процедури завчасної реконфігурації. Поярусне скорочення критичного шляху дозволяє адаптувати процес відображення задач на реконфігуровну обчислювальну структуру до змінюваних умов відображення, в контексті непередбачуваних просторових і часових обмежень реконфігуровної обчислювальної області на ПЛІС.

Таблиця 1.Формальна оцінка прискорення реконфігурації за рахунок видалення повторного завантаження конфігураційних даних

	Час реконфігурації (T^B)	**Обсяг видалення непродуктивних витрат** (ΔR_{remove})
Стандартна послідовність реконфігурації	$T^B = \sum_{j=1}^{K} P_j R_j + \sum_{j=1}^{K} P_j T_j$	-
Завантаження із центральної бібліотеки конфігурацій	$T^B_{rapid_\mathrm{I}} = \sum_{j=1}^{K} R_j^{\mathrm{I}} + \sum_{j=1}^{K} P_j T_j$	$\Delta R_{remove_\mathrm{I}} = \sum_{j=1}^{K} R^{\mathrm{I}}{}_j (P_j - 1)$
Завантаження із локальної пам’яті обчислювального модуля	$T^B_{rapid_\mathrm{II}} = \sum_{j=1}^{K} R_j^{\mathrm{II}} + \sum_{j=1}^{K} P_j T_j$	$\Delta R_{remove_\mathrm{II}} = \sum_{j=1}^{K} P_j R_j^{\mathrm{I}} + \sum_{j=1}^{K} R^{\mathrm{II}}{}_j (P_j$
Зберігання конфігурацій на поверхні ПЛІС	$T^B_{rapid_\mathrm{III}} = \sum_{j=1}^{K} P_j T_j$	$\Delta R_{remove_\mathrm{III}} = \sum_{j=1}^{K} P_j R_j^{\mathrm{I}} + \sum_{j=1}^{K} P_j R_j^{\mathrm{II}}$

На підставі виконаної формалізації (табл. 1) отриманий наступний математичний вираз для визначення часу адаптивного відображення згідно з запропонованим поярусним підходом:

$$T_{G_DAG} = \sum_{j=1}^{K} T_{IO_j} + \sum_{h=1}^{H_1} R_h + \sum_{k=1}^{w} \max(\sum_{h=1}^{H(k+1)} R_h, \{T_h \big| h = \overline{1, H_k}\}) , \qquad (2)$$

де $k = \overline{1, w}$ – номер ярусу, w – кількість ярусів обчислювального алгоритму, $v = \overline{1, H_k}$ – номер вузла на ярусі k, H_k – кількість вузлів на ярусі k.

Час обчислення алгоритму визначається мінімальною кількістю непродуктивного часу, яке складається з часу реконфігурації завдань першого ярусу і витрат на синхронізацію процесів обчислення і реконфігурації.

Моделювання та результати досліджень

Розроблений емулятор реконфігурованої обчислювальної системи (РОС). Емулятор РОС включає наступні блоки, що реалізовані як програмні модулі, які емулюють функції відповідних апаратних блоків: *HardwareSystem* – головний модуль, що емулює функціонал контролера реконфігурації та включає підлеглі модулі: *FPGA* – реконфігурована область; *memory* – локальна пам’ять, що включає функціонал КЕШ пам’яті першого рівня; *bonuses* (підтримка бонусів). Модуль *HardwareSystem* виконує функції пошуку конфігурації задачі – findConfiguration та завантаження задачі в ПЛІС – *load*. Функція *findConfiguration* повертає один з трьох результатів: *TSK_FPGA*, *TSK_MEM*, *TSK_LIB*, значення яких встановлюються на підставі сигналів *hit* або *miss* (вдалого, або невдалого, пошуку) від КЕШ пам’яті першого (КЕШ I) або другого (КЕШ II) рівнів відповідно. Детально архітектура емулятора описана авторами в попередніх роботах авторів [43].

Процес адаптивного відображення змодельовано згідно з математичною моделлю (2). Залежно від умов відображення, які характеризуються певними ресурсами часу, наявністю вільного місця або заздалегідь сконфігурованих обчислювальних ресурсів на обчислювальній поверхні ПЛІС, необхідністю оптимізації поверхні ПЛІС, обирається та чи інша послідовність реконфігурації згідно з визначеними в таблиці 1 математичними моделями. У роботі авторів розглянуто засоби оптимізації реконфігурованих обчислень за критеріями визначеного часу реконфігурації з врахуванням обмежень кристалів ПЛІС, згідно з чим засоби емуляції реконфігуровної обчислювальної системи в динамічному режимі визначають оптимальне місце кешування конфігураційних даних у рамках обмежень реконфігурованих обчислювальних систем, що виникли в процесі обчислень.

Дослідження проводились для серії алгоритмів, поданих графами алгоритмів у ЯПФ. Досліджувались графи алгоритмів з різною кількістю однотипних задач та різним ступенем зв’язності. Досліджувані алгоритми випадковим чином синтезовані на підставі розробленої бібліотеки апаратних реалізацій функціональних ядер, що відповідають певним макрозадачам. Функціональні блоки апаратних задач синтезовані на мові опису апаратури *Verilog* та реалізовані на ПЛІС *Cyclone* II *EP*2*C*35*F*672*C*6 компанії *Altera*. Для верифікації і дослідження часових характеристик функціональних блоків використана плата *AlteraDevelopmentKit DE*2.

На підставі проведених експериментів отримані залежності часу реконфігурації від кількості типів виконуваних задач (рис. 3).

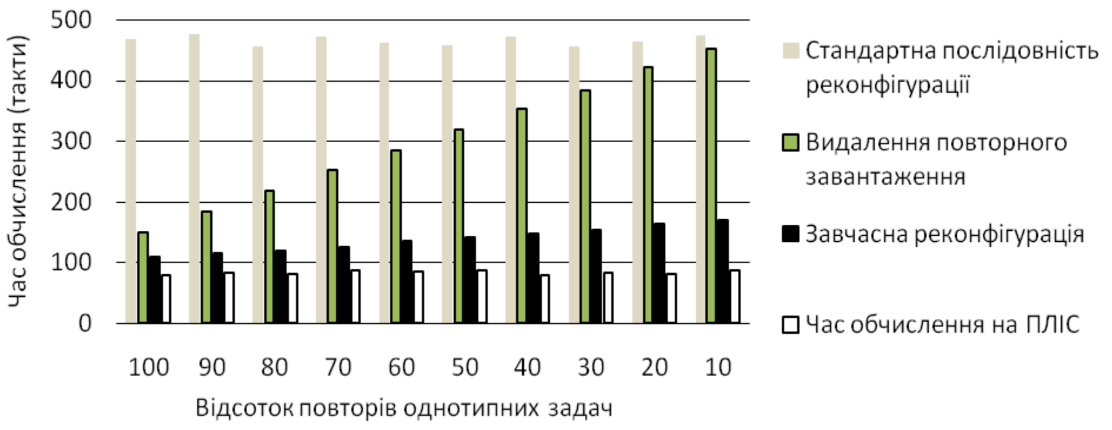


Рис. 3. Дослідження часу реконфігурації від кількості однотипних задач

Механізми повторного використання реконфігурованих ресурсів дозволяють прискорити швидкість реконфігурації в середньому на 63%. При цьому ефективність алгоритмів повторного використання ресурсів залежить від кількості однотипних задач в обчислювальному алгоритмі (рис. 3). З результатів експериментів видно, що завчасна реконфігурація дозволяє видалити практично весь непродуктивний час реконфігурації для будь-яких обчислювальних алгоритмів, незалежно від кількості однотипних задач. Під час дослідження показників прискорення реконфігурованих обчислень за застосування запропонованих засобів отримано, що завчасна реконфігурація в комплексі з видаленням повторного завантаження конфігураційних даних в 2,5 рази зменшує час реконфігурації у порівнянні зі стандартною послідовністю реконфігурації.

Досліджено показник прискорення реконфігурації для алгоритмів з високою щільністю надходження нових задач і навпаки (рис. 4).



Рис. 4. Дослідження коефіцієнту прискорення для видалення повторного завантаження конфігураційних даних

Показник прискорення реконфігурації розрахований як відношення часу стандартної послідовності реконфігурації до часу реконфігурації із застосуванням запропонованих засобів. Взагалі, прослідковується позитивна динаміка збільшення показника прискорення за застосування запропонованих засобів прискорення із видаленням повторного завантаження конфігураційних даних на ПЛІС. Видно, що за сумарними параметрами обчислювального алгоритму і реконфігуровної області коефіцієнт прискорення збільшується. Оптимізація здійснюється за рахунок

застосування копії конфігурацій всіх задач у локальній пам’яті обчислювального модуля. Це дозволяє зберігати апаратні ресурси ПЛІС, зменшувати додаткові витрати часу на створення копій функціональних блоків на поверхні ПЛІС, не зупиняючи при цьому функціональні блоки апаратних задач, що працюють. У цілому, така оптимізація прискорює процес реконфігурації.

Висновки

Запропоновано математичні моделі визначення часу адаптивної реконфігурації, які враховують особливості впливу комунікаційних затримок на швидкодію обчислень, і дозволяють визначити значення критеріїв, що істотно впливають на ефективність реконфігурованих обчислень, а саме: часу реконфігурації, непродуктивних витрат реконфігурації, обчислювальної складності алгоритмів відображення.

Запропоновано формалізацію нового підходу до адаптивного відображення задач на реконфігуроване обчислювальне середовище, який відрізняється від відомих компромісом динамічного і статичного підходів, шляхом завчасної реконфігурації й видалення повторного завантаження конфігурацій функціональних блоків. Це дозволяє скоротити непродуктивні витрати часу і підвищити ефективність реконфігурованих обчислень під час вирішення алгоритмів з великою кількістю однотипних функцій.

Запропоновано спосіб адаптивного скорочення критичного часу виконання алгоритмів, розроблений для реалізації адаптивного відображення, що відрізняється використанням модифікованого методу гілок і границь, що дозволяє оптимізувати процедуру відображення завдань при змінюваних умовах відображення з урахуванням мінімізації часу обчислення і просторових обмежень ресурсів ПЛІС.

Література

1. Al-Wattar A. Efficient On-line Hardware/Software Task Scheduling for Dynamic Run-time Reconfigurable Systems [Text] / A. Al-Wattar, S. Areibi, F. Saffih // Proceeding in 26th International Parallel and Distributed Processing Symposium Workshops & PhD Forum (IPDPSW). – IEEE, 2012. – P 401 – 406.
2. Liu S. Achieving Energy Efficiency through Runtime Partial Reconfiguration on Reconfigurable Systems [Text] / S. Liu, R.N. Pittman, A. Forin, J.-L. Gaudiot // Transactions on Embedded Computing Systems (TECS). – USA,NY, New York, ACM, 2013. – Vol. 12. – P.72:1 – 72:21.
3. Кулаков Ю.О. Розробка методу прискорення реконфігурації в динамічно реконфігурованих обчислювальних системах [Текст] // Ю. О. Кулаков, І. А. Клименко, М. В. Рудницький 2015 // Східно-Європейський журнал передових технологій. – Харків.: УДА залізничного транспорту. – 2015. – №4/4 (76). – С. 25 – 29.
4. AhmedW. Adaptive Resource Management for Simultaneous Multitaskingin Mixed-Grained Reconfigurable Multi-core Processors [Text] /W. Ahmed, M. Shafique, L. Bauer, J. Henkel // Proceeding sof the 9th International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS) (Taiwan, Taipei, 9 – 14 October 2011). – IEEE, 2011. – P. 365 – 374.
5. Клименко І Спосіб управління зернистістю задач в реконфігурованих обчислювальних системах / Клименко І.А // Всеукраїнська наук.-практич. конф. Перспективні напрямки сучасної електроніки, інформаційних та комп'ютерних систем (MEICS-2015) (Україна, Дніпропетровськ, 25-27 листопада 2015). – Дн.: ДНУ ім. О. Гончара, 2015. – С. 117 – 118.
6. Клименко І.А. Оптимізація реконфігурації в динамічно реконфігурованих обчислювальних системах / І.А. Клименко [Текст] // Вісник НТУУ «КПІ». Інформатика, управління та обчислювальна техніка: Зб. Наук. Пр. – К.: Век+, 2014. – №63. – С. 93 – 100.
7. Кулаков Ю.О. Метод оптимізації ярусно-паралельної форми подання задачі для реконфігурованих обчислювальних систем / Ю.О. Кулаков, І.А. Клименко // Електроніка та зв’язок. – К: НТТУ «КПІ», 2014. – Том 19, №4(81). – С. 90 – 96.

References

1. Al-Wattar, A., Areibi, S., Saffih, F. Efficient On-line Hardware/Software Task Scheduling for Dynamic Run-time Reconfigurable Systems // *Proceeding of the 26th International Parallel and Distributed Processing Symposium Workshops & PhD Forum* (*IPDPSW*), China, Shanghai, 21 – 25 May, 2012. – 2012. - pp. 401 – 406.
2. Liu S., Pittman, R.N., Forin, A., Gaudiot J.-L. Achieving Energy Efficiency through Runtime Partial Reconfiguration on Reconfigurable Systems // *Transactions on Embedded Computing Systems* (*TECS*). – 2013. – Vol. 12. – pp. 72:1 – 72:21.
3. Kulakov, Y.O., Klymenko, I.A, & Rudnytskyi, M.V. Development of the reconfiguration acceleration method in the dynamically reconfigurable computing systems. // *Vostochno-Evropeyskiy zurnal peredovyh tehnologiy* [Eastern-EuropeanJournalofEnterprise Technologies]. – №4/4 (76). – 2015. – pp. 25–29. (in Ukrainian)
4. Ahmed W., Shafique, M., Bauer, L. and Henkel,J. Adaptive Resource Management for Simultaneous Multitasking inMixed-Grained Reconfigurable Multi-core Processors // *Proceedings of the 9th International Conference onHardware/Software Codesign and System Synthesis* (*CODES+ISSS*), Taiwan, Taipei. – 2011. – pp. 365 – 374.
5. Klymenko I. The method of management granularity of the tasks in the reconfigurable computer systems // *Proceeding soft he Ukrainian scientific-practical conference Perspective direction sofmodern electronics, information and computer systems (MEICS-2015)* (Ukraine, Dnipropetrovsk, 25 – 27 November 2015). – DNU, 2015. – P. 117 – 118.
6. Klymenko, I.A. The effectiveness analysisof resources management in reconfigurable computer systems // *Visnyk NTUU "KPI". Informatyka, upravlinnia ta obtchislyuvalna technika: Zb. Nauk. Pratsc*. – № 62. – 2015. – pp. 11 – 21.(in Ukrainian)
7. Kulakov Y.O., Klymenko I.A. The optimization method of a macro dataflow graph for reconfigurable computing systems // *Electronika I zvjazok* [Electronics and Communication]. – Vol. 19, № 4(81). – 2014. – pp. 90 – 96. (in Ukrainian).

RESUME

I.A. Klymenko

The formalization of adaptive tasks mapping in the reconfigurable computing systems on FPGAs

This article resolves the problem of reducing time overheads of task mapping process on dynamically reconfigurable computing structure of FPGA. This is achieved by reducing communication delays during setup the computing environment to meet the requirements of each task, which is resolving.

The proposed mathematical models for determining the adaptive reconfiguration time, take into account the influence of the communication delay on the performance of the reconfiguration, allow determining the values of criteria that have a significant impact on the effectiveness of reconfigurable computing system, namely the reconfiguration time, reconfiguration time overheads, the computational complexity of mapping algorithms. Based on these models, we propose a formalization of a new approach of adaptive task mapping on a reconfigurable computing environment. Our method differs from the known by usage both, dynamic and static approaches; namely pre-reconfiguration and disabling re-loading of the configuration of functional blocks. It allows reducing time overheads and improving the efficiency of reconfigurable computing for resolving tasks with frequent repetitions of the same type of functions.

The proposed method of adaptive reducing of critical execution time of algorithms to implement the adaptive mapping process is designed. We use the modified branch and bound method, which allows to optimize tasks mapping process with changeable mapping conditions, considering the minimization of the computation time and spatial constraints of FPGA resources.

І.А. Клименко

Формалізація адаптивного відображення задач у реконфігурованих обчислювальних системах на ПЛІС

У статті вирішується проблема зменшення непродуктивного часу відображення задач на динамічно реконфігуровану обчислювальну структуру на ПЛІС за рахунок зменшення комунікаційних затримок під час налаштування обчислювального середовища згідно з вимогами кожної вирішуваної в певний момент часу задачі.

Запропоновано математичні моделі визначення часу адаптивної реконфігурації, що враховують вплив комунікаційних затримок на швидкодію реконфігурації й дозволяють визначити значення критеріїв, що істотно впливають на ефективність реконфігурованих обчислень, а саме: часу реконфігурації, непродуктивних витрат на реконфігурацію, обчислювальної складності алгоритмів відображення. На основі цього запропоновано формалізацію нового підходу до адаптивного відображення завдань на реконфігуроване обчислювальне середовище, який відрізняється від відомих компромісом динамічного і статичного підходів, шляхом попередньої реконфігурації і видалення повторного завантаження конфігурацій функціональних блоків. Це дозволяє скоротити непродуктивні витрати часу і підвищити ефективність реконфігурованих обчислень при вирішенні завдань з частими повтореннями однотипних функцій.

Запропоновано новий спосіб адаптивного скорочення критичного часу виконання алгоритмів, який розроблений для реалізації адаптивного відображення і відрізняється використанням модифікованого методу гілок і границь, що дозволяє оптимізувати процедуру відображення алгоритму при змінюваних умовах відображення з урахуванням мінімізації часу обчислення і просторових обмежень ресурсів ПЛІС.

Надійшла до редакції 22.06.2016